# 商铺关联分析：

1. 使用Apriori原理得到商铺的最高M元频繁集
2. 使用Apriori原理得到商铺的每个频繁集中蕴含的关联规则
3. 基于关联规则得到商铺之间的有向依赖图（邻接表）
4. 将邻接表转化成邻接矩阵，然后转化为page rank转移矩阵
5. 根据page rank转移矩阵得到每个商铺的评分

涉及到的数据库表**mac\_portal：**

mac

portalId

## 使用Apriori原理得到商铺的最高M元频繁集

1. **生成商铺共现清单：**

清单结构：

|  |  |
| --- | --- |
| {portalId列表} | 出现次数counter |

生成方法：

* 生成用户mac表mac：

Id// 自增id

mac

insert into mac(mac) select distinct mac from mac\_portal

* 并行生成清单结构，每个线程获取一部分mac，并执行：

select \* from mac\_portal where mac in {“xxxx”,”yyyy”....} group by mac, portalId

对相同的mac访问过的portalId组织成portalId列表，存储进清单结构

1. **进行频繁集搜索**

初始化频繁集列表，每个频繁集只有一个portalId

I = 1;

While(i <= M) {

在清单中搜索每个频繁集出现的次数，并计算、保存其支持度，记为S

剔除频繁集列表中支持度小于阈值的频繁集

把频繁集列表中的频繁集组合出更大的频繁集，合并出的频繁集大小为i+1

++i

}

**该算法的详细设计：**

数据结构设计：

频繁集 List<Integer>，内容按照升序排列

频繁集列表Map<Integer, List<List<Integer>>> key是频繁集的长度

商铺共现清单 Map<Set<Integer>, Integer> key是portalId的集合，value是出现次数

流程：

向频繁集列表中加入初始化频繁集，每个频繁集中只有一个portalId

counter = 1

do {

在频繁集列表中找到key=counter的频繁集子列表subList

检查subList中每个频繁集在商铺共现清单中出现的次数

删除掉subList中支持度小于阈值的频繁集

// 频繁集合并

++counter

在频繁集列表中建立新的频繁集subList2

For(挑选subList中的两个频繁集)

调用combine(s1, s2) s1s2是挑选的两个频繁集

} while(counter <= M)

combine(s1, s2) {

把s1，s2进行从小到大排序

If(s1和s2去除最后一个元素后的部分是相同的) {

把s1 s2合并，放入到subList2

}

}

## 使用Apriori原理获得每个频繁集中的关联规则

关联规则是频繁集的一个划分

对于每一个频繁集{p1, p2, p3}

对于p1,p2->p3的可信度，计算方式是S(p1p2p3)/S(p1p2)

假如p1,p2->p3的可信度小于阈值，则不用考虑对于频繁集划分左部是p1p2子集的情况

**该算法的详细设计：**

数据结构设计：

关联规则表 Map<List<Integer>, List<Integer>> key是关联规则的前部，value是规则的后部

While(遍历频繁集列表) {

对于每个频繁集F(频繁集一定大小大于等于2)：

初始化：选一个元素作为右部，剩下的作为左部，写入temp关联规则表

counter = 1

do {

for(遍历每一个关联规则) {

计算这个关联规则的可信度

删除可信度小于阈值的关联规则

}

把temp关联规则表中剩下的关联规则加入到关联规则表中

++counter

把temp关联规则表中的右部收集起来，组成subList，新建temp

新建subList2

For(挑选subList中的两个频繁集)

调用combine(s1, s2) s1s2是挑选的两个频繁集

subList2中的频繁集都是新的关联规则的右部，根据F得到关联规则的左部，添加到temp中去

} while(counter < 频繁集的大小)

}